NoSQL Proof-of-Concept Proposal – Dota2 Game Replay Analysis

Team name: PSG.LGD

Yichun Yan
Ziwei Jiang
Yifan Li
Weiqi Wang

October 2, 2019

Contents

1 Potential Datasets

Our team aims at exploring data about a popular and long-lived computer game, Dota2. We will mainly use two datasets:

1. Dota2 match result dataset
2. Dota2 replay dataset

The first dataset is the snapshot of the final state of each player and the whole game, containing only important information. The second one is a binary format file which can be executed by Dota2's client to reproduce everything happened in the game.

The datasets can be retrieved from the following resources:

1. Valve's Dota2 replay servers
2. Valve's official API

We will get the match results data from Valve's official API. It also provides a key, which we can use to retrieve a replay from the replay server.

In all, we will have two different kinds of data for our further analysis, relatively large-scale but coarse-grained match results data, and very fine-grained replays data. Considering the file size of one replay, we decide to include recent professional games and randomly select some public games for the replay data. WE will mention how we acquire the data in Data ETL section.

2 Five Vs of Datasets

To assess the adequacy of adopting big data to our project, we firstly identify five Vs characteristics of our dataset.

2.1 Volume

Replays are one of the datasets we plan to use, which are the full records of finished matches. The size of one replay is estimated to be about 60MB. We plan to collect 3000 replays whose size is about 180GB. As for match result data which is JSON-format data, the size of each result is about 5KB. We plan to collect 10,000,000 match results whose size is about 50GB. Moreover, if we desire more data in future implementation, we can always access it as long as Dota 2 still has players.

2.2 Velocity

According to the statistic data in August 2019 from Steam which is a video game digital distribution platform, the average number of players is 467,148.3 players and the peak number is 826,690. A large number of players makes both the replay data and result data high-velocity. Although in our project, the velocity depends on the velocity of the API request, this statistic data reveals that it's potential to be high-velocity.

2.3 Variety

From the APIs mentioned above, we can access JSON format files, the match result data, and binary format files, the replay data.

2.4 Veracity

The overall quality of our datasets is good because we access most of them from the official API. But there is still a few noises. For example, the time of some games is too short, which makes these game lack of representativeness. Thus, we plan to implement data validation to filter the undesired matches.

2.5 Value

Our datasets are useful for a relatively long time. Because we plan to collect 1/3 professional matches and 2/3 public matches. The professional matches are durable since we can always extract information about professional teams and players from them. As for the public games, they are valuable as long as they are not too stale, like more than 2 years ago. In the data validation step, we will set a filter on the date for public games which makes our datasets valuable.

3 Potential Business Questions

As mentioned before, we can collect 2 kinds of data, one is match results and one is replays. By analyzing them, the following questions might be answered.

- Easy questions:
  - Who is the hero gaining gold/XP fastest in 15 minutes in public games/professional games?
  - Who is the hero having most kills/assists/heals/deaths in public games/professional games?
  - What is the most purchased item in public games/professional games?
  - Who is the hero having most bad-manner players (players who are AFK or disconnected)?
  - Who is the most popular hero (the hero who has the highest pick rate) in public games/professional games?
  - Who is the hero having the highest ban rate in professional games?
  - who is the hero having the most ban/pick rate in professional games?
  - How long does a game cost on average in public games/professional games?
  - Which item is used most in public games/professional games?
- Moderately challenging questions:
  - How are the benefits gained from buybacks in public games/professional games
  - How is the vision in public games/professional games?
  - When does the first team battle happen in public games/professional games?
  - Which is lineup having the highest win rate in public games/professional games?
  - Which is the most popular lineup (lineup which has the highest pick rate) in public games/professional games?
  - which is the most common ban-pick combo in professional games?
  - Who is the hero changed most in win/pick/ban rate after releasing a new version of Dota2?
  - Who is the hero having the highest winning rate in each lane?
- Challenging questions:
  - How is the winning-losing relationship between Dota 2 professional teams? Can we use a graph to visualize it?
  - Does there exist a regular farming path for some professional players?
  - Does there exist a correlation between the time of the first blood and the time of the entire game?
  - Does there exist a correlation between the gold/XP source of a hero and its win rate?
  - Does there exist a correlation between the distribution of the economy and the result of the game?
  - Does there exist a correlation between economic development and the time of the first team battle?

Moreover, we can use the answers above to analyze the difference between public games and professional games as well as the difference between the blind-pick game and the draft-pick game.

4 Data ETL

4.1 Extraction

Valve, the company that develop the game, initially try to provide all the players with easy APIs to access the data of the game. But due to the increasing stress on its data servers, many of its APIs are shut down. At the same time, the documentation seems to be never updated since it's created, which makes it much more difficult for us to collect the data. For example, the API which returns a bunch of the match results given a starting match id is not usable anymore. Another API which returns a key that is critical for construct the URL to download replay of that game is not working as well.

But luckily, we found some hints from the developer's forum of Dota2. We can use the starting sequence number, which works similarly as the match id, to get a series of game results. And by calling a third party API from OpenDota, we can get the important key to construct the URL to download replays again.

The specific steps of gaining our data are as following:

1. Match Results:
   - Use GetMatchHistoryBySequenceNum API to get the match ids. We will need a a field called `start_at_match_seq_num` to specify the starting match sequence number of the results.
   - Then we can extract the last sequence number of the results as the `start_at_match_seq_num` for the next call. By doing this iteratively we can enlarge our dataset for our first kind of data.
2. Replays:
   - Get the match ids of recent professional games from the last step.
   - Use the those ids via OpenDota API to get the information we need (`cluster` and `replay_salt`) for retrieving replays from Valve's replay server
   - Construct links in this format: `http://replay<cluster>.valve.net/570/<match_id>_<replay_salt>.dem.bz2` to get the replays.

4.2 Transformation

The match result is in JSON format so we can almost directly store it. The information it contains are:

- Players information:
  - Account id
  - Player slot
  - The hero he use in this match
  - The items he possess at the end of the match
  - The player's kills/deaths/assists
  - The player's leaver status (whether AFK/disconnected or not)
  - The gold he possess at the end of the match
  - The amount of last-hits and denies the player got during the match
  - The average gold/XP he gained per minute
  - The total gold he spent during the match
  - The amount of damage the player dealt to heroes/towers
  - The amount of health the player had healed on heroes
  - The player's level at match end
  - A list detailing a player's ability upgrades, including the ability and the upgrading time
  - Additional playable units owned by the player and its items
- The season the game was played in
- The winner team of the match
- The length of the match

- Unix timestamp of when the match began
- The matches unique ID
- A sequence number, representing the order in which matches were recorded
- The tower status of both teams
- The barracks status of both teams
- The server cluster the match was played upon
- The time in seconds since the match began when first-blood occurred
- The mode of the match
- The amount of human players within the match
- The league that this match was a part of
- The number of thumbs-up/thumbs-down the game has received by users
- A list of the picks and bans in the match, if the game mode is Captains Mode

On the other hand, the replay data is completely unstructured `.dem.bz2` binary file, data transformation must be done before we can store the data in our database.

Firstly, we can decompress the `.bz2` file with `org.apache.commons.compress.compressors.bzip2` package in Java, which will give us a `.dem` file:

Next, we can utilize clarity, an open source Dota2 replay parser, to extract useful information from the `.dem` file.

It cannot be achieved by a single click, though. We have to write a lot of code to invoke its function. What's more, clarity does not provide a detailed documentation, instead there are only some examples which forces us to iteratively attempt and learn the usage of this tool.

Based on our exploration, the following data will be available:

- Player name, id, team formation and hero choice
- Detailed log of the game, including a hero:
  - deals damage to another one
  - heals another one
  - receives/loses a buff/debuff
  - kills another one
  - uses his ability
  - uses an item
  - buys an item
  - receives/loses some gold
  - gains some XP
  - buys back (spending money in order to instantly re-spawn)
- Spawn/death of heros and NPCs

These information will be organized in a Document and store to our database.

Most fields in our datasets are related to some of the above-mentioned business questions. They are all important. We will generally validate them, for example, fields like the HP, XP, total gold cannot be negative.

What's more important is to filter out invalid data on the game level. Some of the games lasts for only two or three minutes, because some players are disconnected and others just quit the game very quick. Others are practicing games where most players are computer bots. We will completely drop these game records.

4.3 Loading

4.3.1 NoSQL Storage Technology

As we mentioned above, we have two datasets - one is the match results data in JSON format, which is a kind of document that can be encoded using a text-based encoding scheme. The replay data is binary files which is

inappropriate for document NoSQL database to store. But after we parsing those replay data, we can gather useful information from replay data in Document format.

Therefore, we choose to use MongoDB as our primary NoSQL storage technology.

MongoDB is a popular NoSQL storage device that can store high volume, high velocity, high variety Big Data datasets. It's a document storage device, which make storing semi-structure document-oriented data such as JSON much easier. It is an open source NoSQL database, so we are free to use it.

Also, MongoDB support partial update. This will help us aggregate values in future games that we would potentially use.

4.3.2 Docker

In order to combine different needs, programming languages and developing environment of many technologies, we will adopt Docker to our project. Using Docker, we can deploy our application into any environment.

Docker is a container technology that provide consistency. It's open source and there are many open source applications that have been made in Docker Hub. Docker Hub provides official or verified images, so it's safe to use.

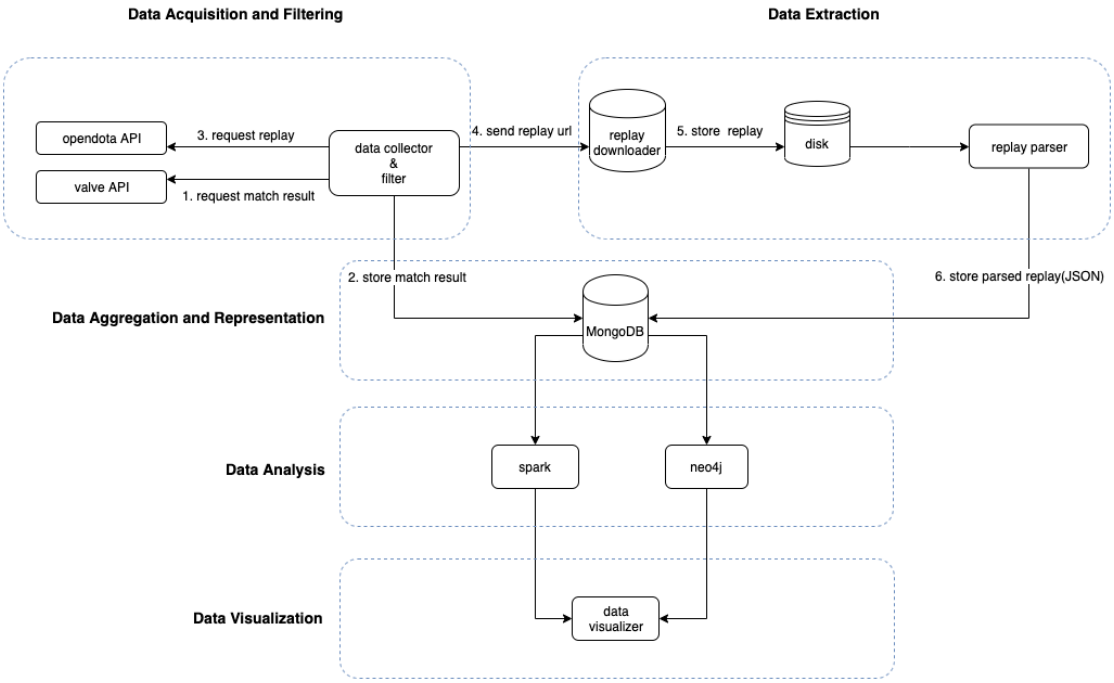
In our project, we will build Dockerfile to let it connect with our project. We will use Scala as programming language, and we will get the match results data from Dota2's official API. MongoDB will be used as database. And then we will put all of this into a Docker container.

4.3.3 AWS (Amazon Web Service)

As we mentioned before, we need 50GB to store JSON file, and we need 180GB to store replay data, and the scale would be extended if we add more data in the datasets. In order to have enough capacity to store datasets, we will use AWS (Amazon Web Server) to deploy our project on cloud server. Running Docker on AWS will provide us a open source and free way to build and run distributed application and provide a reliable and easy way to scale out.

AWS provides numbers of official ways to deploy docker on AWS, such as Amazon ECS, which is a highly scalable, high-performance container orchestration service to run Docker containers on the AWS cloud. After we package datasets and analytics packages into Docker, we will deploy Docker to Amazon ECS.

5 Overview of our Architecture



6 Potential Challenge

There are several aspects where we may have potential challenge.

During the environment setup:

- We are trying to build a docker containerized application, but none of our team members have previous experience
- We do not have experience about using AWS or other cloud services, neither

In the data ETL phase:

- Download replays from the server can be slow, the links may not work
- Professional games are relatively rare, so more efforts will be needed to find and retrieve the professional replays
- Both APIs have call limits, even though we can control our request frequency, some request may still fail
- Some downloaded replay files may be broken so that no information can be extracted, or they may even cause exceptions in the pipeline
- The open source replay parser does not have a documentation, thus some of our parsing code may cause unexpected exceptions when there are some corner case

In the data analysis phase:

- Lots of joins and aggregations between the two datasets will be involved
- We must quantitatively define some generally used concepts which are often qualitatively used, for example, the measurement of vision, team battle, etc. An inappropriate definition can make the whole conclusion meaningless